Proof
From Theorem 14, it is sufficient to show that it is computable whether there exists a cycle C for which $l(C) = 0$ *and* $lhl(C) = 0$. Let $\{C_1, \ldots C_j, \ldots C_t\}$ be a composable subset of the basic cycles of G. Then consider a cycle C composed from y_j instances of C_j, for each j, $1 \leqslant j \leqslant t$. Then

$$l(C) = \sum^{t} y_j \, . \, l(C_j)$$

$$lhl(C) = \sum^{t} y_j \, . \, lhl(C_j) \, .$$

In general there are a finite number of alternative expressions for $l(C_j)$ and $lhl(C_t)$, each on the lines of the proof of Theorem 13. For each combination of alternatives the simultaneous equations

$$l(C) = 0$$

$$lhl(C) = 0$$

take the form of Diophantine equations (2) and the Diophantine result 5 shows it to be computable whether a solution exists.

If one of these sets of equations proves to be solvable, then the algorithm for deciding that also leads readily to one solution at least. This means that one cycle with the desired properties is known. Suppose that N is a non-terminal in the cycle, say $N \Rightarrow u_s N v_s$, where u_s and v_s are terminal strings. Then choose any derivation $S \Rightarrow u'Nv'$ for any terminal u' and v', and a non-cyclic derivation $N \Rightarrow s_N$. One can then show that an infinity of terminal strings generated by the cycle is $u'u_s^n s_N v_s^n v'$, $n \geqslant 1$, whose upper and lower bounds are the same as those for $u'u_s s_N v_s v'$, where $n = 1$. Thus one suitable interval $[L_1, L_2]$ can be found if any exist at all.

Theorem 16
It is possible to find a smallest interval, if any, in which G is infinitely stack-bounded.

Proof
This theorem just brings together the results of Theorems 11 and 15.

References

GOODWIN, D. T. (1977). Conditions for Underflow and Overflow of an Arithmetic Stack, *The Computer Journal*, Vol. 20, pp. 56-62.

MORDELL, M. J. (1969). *Diophantine Equations*, pp. 30-1. London: Academic Press.

Book reviews

Software Portability: an Advanced Course, edited by P. J. Brown, 1977, First paperback edition 1979; 328 pages. (*Cambridge University Press*, £5·50)

The text is drawn from an advanced course on software portability given at the University of Kent at Canterbury in Spring 1976 and comprises the copious session material of the individual speakers. The course was sponsored by SRC under the auspices of the Informatics training group of the EEC Scientific and Technical Research Committee. Speakers were drawn from both sides of the Atlantic and appeared very much as a listing in 'who's-who' in Software Portability.

The book is divided into eight 'parts' each being subdivided into chapters representing individual speaker's contributions. Each chapter has appended a list of references and these define the state-of-the-art at the time of writing. Parts 1 and 2 are an introduction which outlines the format of the text and a basic concepts review which is really a justification of what follows. Part 3 concerns itself with mechanisms which affect and aid portability. It includes descriptions of the roles played by verifiers and filters, computer architecture and microprogramming, and microprocessors. The problems associated with portable compilers and the translation between high level languages are discussed. Part 4 considers pragmatics, the practical minutiae which so greatly affect portability as a useful tool. Considerations include engineering, the system interface, performance and optimisation. This 'part' is very practical and informative. Part 5 has a novelty as well as great general use. It addresses itself to legal aspects and provides a valuable insight into the legal standing of 'information'. The session was driven by the portability concept but turns out to be of more general interest. In part 6 a number of case studies are presented, ranging from the portability of ALGOL 60, through SNOBOL 4, BCPL, commercial software, data, FORTRAN and GENESYS, an operating system, and NAG's approach, to the view of a manufacturer (ICL). The experiences recounted certainly ring true to the reviewer. In Part 7 two of the leading lights in the field (Waite and Griswold) combine to discuss current research in the area and take the possibly dangerous step of predicting the future. Part 8 presents two study group reports on portability, those of CNRS/SRC and EEC.

The text is a wide ranging survey of the nature, problems and mechanisms involved in software portability and succeeds in this area. Since it is formed of a series of papers from different contributors the book suffers to a certain degree because of variations in style, depth and general presentation (including typesetting); however the effect is not as bad as that experienced in many similarly organised texts.

The fragmentation of the book does have an advantage: each chapter is largely self contained and therefore the text easily lends itself as a reference document with respect to the different facets of portability. In general the level of the material is at least final year undergraduate. The contents will be of interest to the software industry at large, since many practical lessons may be learnt from within its covers, without the pain of finding out on a 'real' system.

ROY NEWTON (Middlesbrough)

The Challenge of Microprocessors by M. G. Hartley and Anne Buckley, 1979; 200 pages. (*Manchester University Press*, £7·95)

Hartley and Buckley are the editors of the *International Journal of Electrical Engineering Education*, and this book consists mainly of papers from a special issue of this journal. It is divided into four main sections: Overall Perspective; General Laboratory Activity; Laboratory Experiments; and Courses and Services. Unfortunately, the editors seemed unable to find sufficient material about microprocessors and have padded out the book with chapters on the use of programmable calculators and a description of a large computerised education system, which are totally unrelated to the rest of the book.

An attempt is made to provide a coverage of a wide range of microprocessor topics, which succeeds only in generating confusion. There is little coherence between the chapters, with the result that some are aimed at educators who are totally unfamiliar with microprocessors, while there are three chapters on microprogramming and bit-slice processors, which seem to be aimed at budding minicomputer designers. The typical educational institution approach of costing equipment merely by the value of the electronic components is noticeable, and highlights the lack of information about the commercial side of microprocessors, such as the development costs of hardware and software, project management and so on.

There is little in this book to commend it to the general reader, and since it does not aim at an identifiable part of the microprocessor arena, or of the educational spectrum, it can also not be recommended to any specific audience.

HARLEY QUILLIAM (Guildford)